Stateful Apps в Kubernetes. Как мы работаем с Persistent Data

Владислав Клименко



v2



Введение



ALTINITY
CLICKHOUSE
OPERATOR

for Kubernetes

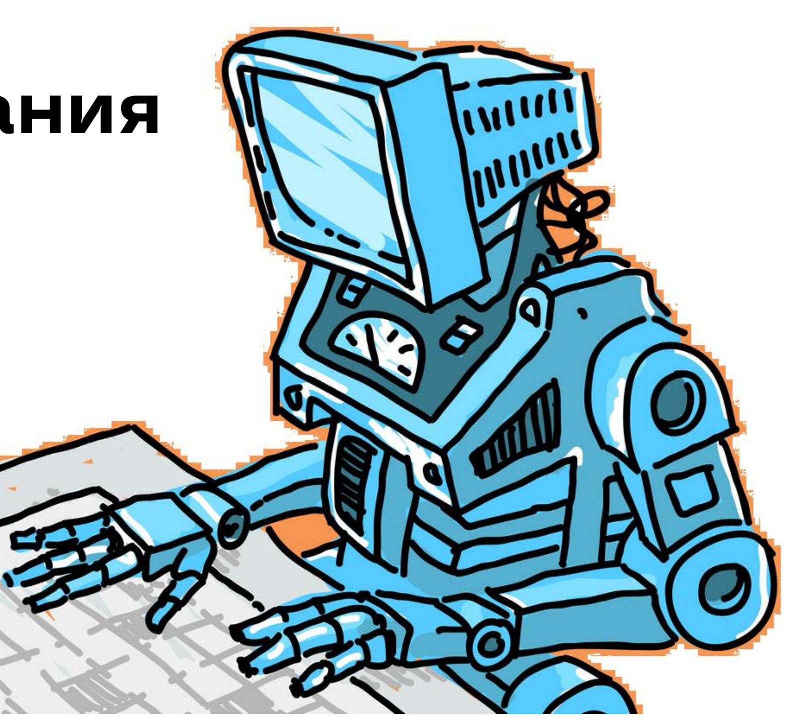
Altinity





- **Помощник DevOps'a**
- **Кодифицированные знания**







ClickHouse Operator creates, configures and manages ClickHouse clusters running on Kubernetes.

release v0.14.0 | PASSED | docker pulls 356k | go report A+ | Go v1.13 | issues 44 open | tag v0.14.0

Features

The ClickHouse Operator for Kubernetes currently provides the following:

- Creates ClickHouse clusters based on Custom Resource specification provided
- Customized storage provisioning (VolumeClaim templates)
- Customized pod templates
- Customized service templates for endpoints
- ClickHouse configuration and settings (including Zookeeper integration)
- Flexible templating
- ClickHouse cluster scaling including automatic schema propagation
- ClickHouse version upgrades
- Exporting ClickHouse metrics to Prometheus



https://github.com/Altinity/clickhouse-operator









ALTINITY.CLOUD

Fully Managed ClickHouse for the Amazon Cloud by the Enterprise Experts



Create Prod-Ready Clusters in Any AWS Region with Ease

Spin up clusters, connect, and start working. Vertical and horizontal scaling? The Altinity.Cloud cluster manager makes it a snap. Zookeeper? You'll forget it's there. High availability? Altinity.Cloud has multi-AZ operation and automatic backup. Upgrade? Altinity.Cloud does it automatically without interrupting service. Monitoring? Dashboards are built-in and ready to use.

Altinity.Cloud runs in the region your applications need, not the region that's convenient for us. Pick any Amazon region and we'll make it work. Active regions are available for immediate deployment. We can add new regions in couple of days, so don't be shy.



ClickHouse Operator creates, configures and manages ClickHouse clusters running on Kubernetes.

release v0.14.0 | PASSED | docker pulls 356k | go report A+ | Go v1.13 | issues 44 open | tag v0.14.0

Features

The ClickHouse Operator for Kubernetes currently provides the following:

- Creates ClickHouse clusters based on Custom Resource specification provided
- Customized storage provisioning (VolumeClaim templates)
- Customized pod templates
- Customized service templates for endpoints
- ClickHouse configuration and settings (including Zookeeper integration)
- Flexible templating
- ClickHouse cluster scaling including automatic schema propagation
- ClickHouse version upgrades
- Exporting ClickHouse metrics to Prometheus



Часть 1. Persistent



ClickHouse Operator creates, configures and manages ClickHouse clusters running on Kubernetes.

release v0.14.0 | PASSED | docker pulls 356k | go report A+ | Go v1.13 | issues 44 open | tag v0.14.0

Features

The ClickHouse Operator for Kubernetes currently provides the following:

- Creates ClickHouse clusters based on Custom Resource specification provided
- Customized storage provisioning (VolumeClaim templates)
- Customized pod templates
- Customized service templates for endpoints
- ClickHouse configuration and settings (including Zookeeper integration)
- Flexible templating
- ClickHouse cluster scaling including automatic schema propagation
- ClickHouse version upgrades
- Exporting ClickHouse metrics to Prometheus



Cluster

Load Balancer Service

Replica Service

Replica Service

Replica Service

Replica Service

Stateful Set

Pod

Persistent Volume Claim

Persistent Volume

Persistent Volume Claim

Persistent Volume

Per-replica Config Map

Shard 1 Replica 1

Shard 1 Replica 2

Shard 2 Replica 1

Shard 2 Replica 2

User Config Map

Common Config Map



Persistent Storage

- Cloud storage
  - AWS
  - GKE
  - Other cloud providers
- Local storage
  - emptyDir
  - hostPath
  - Local
- CSI
- Provisioners



Рассмотрим через призму users & use-cases



Level:

Начальный. Cloud storage

Описание:

Случайно поменял одну строчку, и всё сломалось!

Запрос:

Верните всё назад!



Persistent Storage – cloud storage

- AWS
- GKE
- Other cloud providers

+ просто

Load Balancer Service
Replica Service
Replica Service
Replica Service
Stateful Set
Pod
Persistent Volume Claim
Persistent Volume
Per-replica Config Map
Shard 1 Replica 1
Shard 1 Replica 2
Shard 2 Replica 1
Zookeeper-0
Zookeeper-1
Zookeeper-2
Zookeeper
Cloud
Storage



```yaml
apiVersion: "clickhouse.altinity.com/v1"
kind: "ClickHouseInstallation"
metadata:
  name: "simple-02"
spec:
  configuration:
    clusters:
      - name: cluster1
      - name: cluster2
```



```yaml
apiVersion: "clickhouse.altinity.com/v1"
kind: "ClickHouseInstallation"
metadata:
  name: "simple-02"
spec:
  configuration:
    clusters:
      - name: cluster1
```



```yaml
apiVersion: "clickhouse.altinity.com/v1"
kind: "ClickHouseInstallation"
metadata:
  name: "simple-02"
spec:
  configuration:
    clusters:
      - name: cluster1
      - name: cluster2
```



```yaml
apiVersion: "clickhouse.altinity.com/v1"
kind: "ClickHouseInstallation"
metadata:
  name: "simple-02"
spec:
  configuration:
    clusters:
      - name: "shard1-repl1"
        layout:
          shardsCount: 2
          replicasCount: 2
```



```yaml
apiVersion: "clickhouse.altinity.com/v1"
kind: "ClickHouseInstallation"
metadata:
  name: "simple-02"
spec:
  configuration:
    clusters:
      - name: "shard1-repl1"
        layout:
          shardsCount: 1
          replicasCount: 2
```



```
apiVersion: "clickhouse.altinity.com/v1"
kind: "ClickHouseInstallation"
metadata:
  name: "simple-02"
spec:
  configuration:
    clusters:
      - name: "shard1-repl1"
        layout:
          shardsCount: 2
          replicasCount: 2
```



Cluster

Load Balancer Service

Replica Service

Replica Service

Replica Service

Replica Service

Stateful Set

Pod

Persistent Volume Claim

Persistent Volume

Persistent Volume Claim

Persistent Volume

Per-replica Config Map

Shard 1 Replica 1

Shard 1 Replica 2

Shard 2 Replica 1

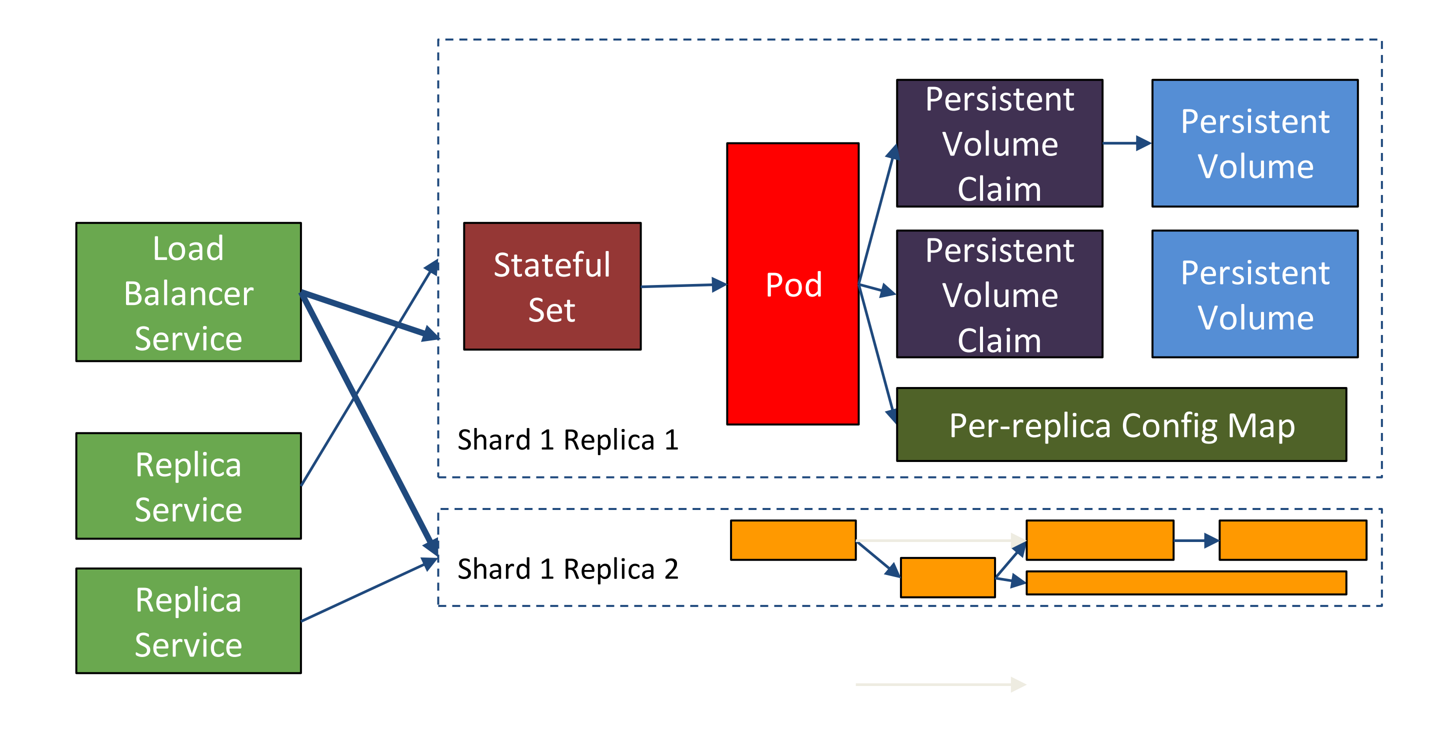
Shard 2 Replica 2

User Config Map

Common Config Map



Cluster



User Config Map

Common Config Map

Cluster

Load Balancer Service

Replica Service

Replica Service

Replica Service

Replica Service

Stateful Set

Pod

Persistent Volume Claim

Persistent Volume

Persistent Volume Claim

Persistent Volume

Per-replica Config Map

Shard 1 Replica 1

Shard 1 Replica 2

Shard 2 Replica 1

Shard 2 Replica 2

User Config Map

Common Config Map



Что будем делать?

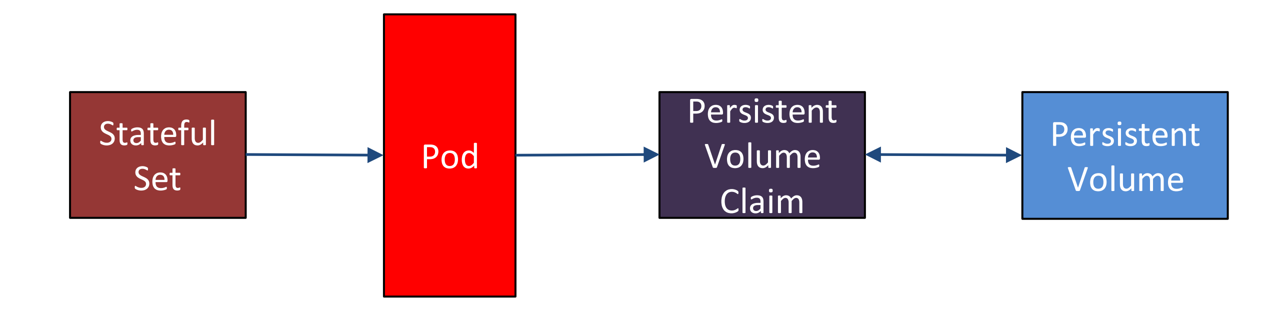


```yaml
apiVersion: v1
kind: PersistentVolume
spec:
  persistentVolumeReclaimPolicy: Retain
```

И ещё немного кода вокруг этого

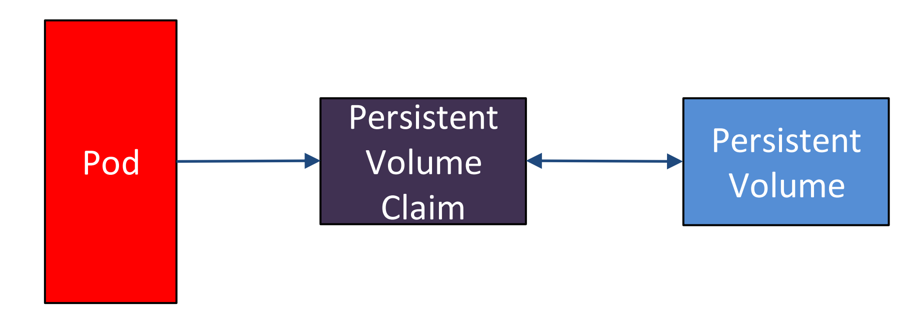


Delete


Stateful Set
Pod
Persistent Volume Claim
Persistent Volume

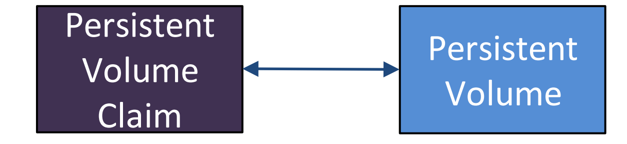

HL
HighLoad++
Весна 2021

Delete


Pod
Persistent Volume Claim
Persistent Volume


HL
HighLoad++
Весна 2021

Delete


Persistent Volume Claim
Persistent Volume


HL
HighLoad++
Весна 2021

Delete

Persistent Volume

Давайте посмотрим на него

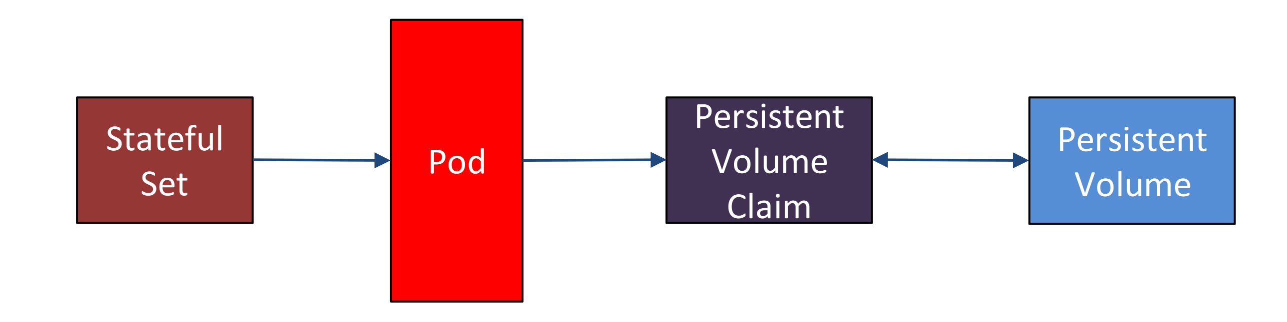


Restore

```
apiVersion: v1
kind: PersistentVolume
spec:
  awsElasticBlockStore:
    fsType: ext4
    volumeID: aws://eu-north-1a/vol-0705ea080f82a429c
  capacity:
    storage: 7Gi
  claimRef:
    apiVersion: v1
    kind: PersistentVolumeClaim
    name: pvc-2
    namespace: dev
    resourceVersion: "52430830"
    uid: 2541da41-e996-4757-8f21-cb4a0f23de4b
  persistentVolumeReclaimPolicy: Retain
  storageClassName: kops-ssd-1-17
```



Restore



Переставить ссылку на PVC



Cluster

Load Balancer Service

Replica Service

Replica Service

Replica Service

Replica Service

Stateful Set

Pod

Persistent Volume Claim

Persistent Volume

Persistent Volume Claim

Persistent Volume

Per-replica Config Map

Shard 1 Replica 1

Shard 1 Replica 2

Shard 2 Replica 1

Shard 2 Replica 2

User Config Map

Common Config Map



А для упрощения лучше сразу сделать правильный StorageClass



```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: kops-ssd-1-17-retain
parameters:
  encrypted: "true"
  type: gp2
provisioner: kubernetes.io/aws-ebs
reclaimPolicy: Retain
volumeBindingMode: WaitForFirstConsumer
```



```yaml
apiVersion: "clickhouse.altinity.com/v1"
kind: "ClickHouseInstallation"
metadata:
  name: "simple-02"
spec:
  configuration:
    clusters:
      - name: cluster1
      - name: cluster2
```



Восстановление данных:

Возможно в полном объёме



Level:

MOAR PERFORMANCE!!!

Local storage

Описание:

Иногда пропадают куски данных

Запрос:

Верните всё назад!

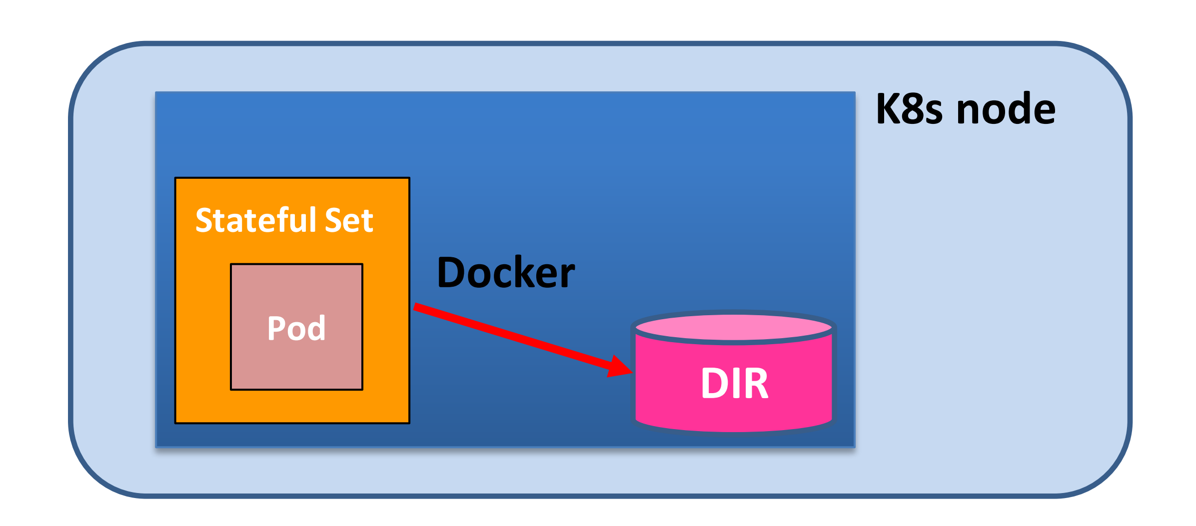


Persistent Storage – local storage

emptyDir – Docker-internals

+ performance

- persistence





Восстановление данных:

50/50

Есть реплики

Нет реплик



Level:

MOAR PERFORMANCE!!!
Local storage

Описание:

Иногда пропадают куски данных

Запрос:

Верните всё назад!

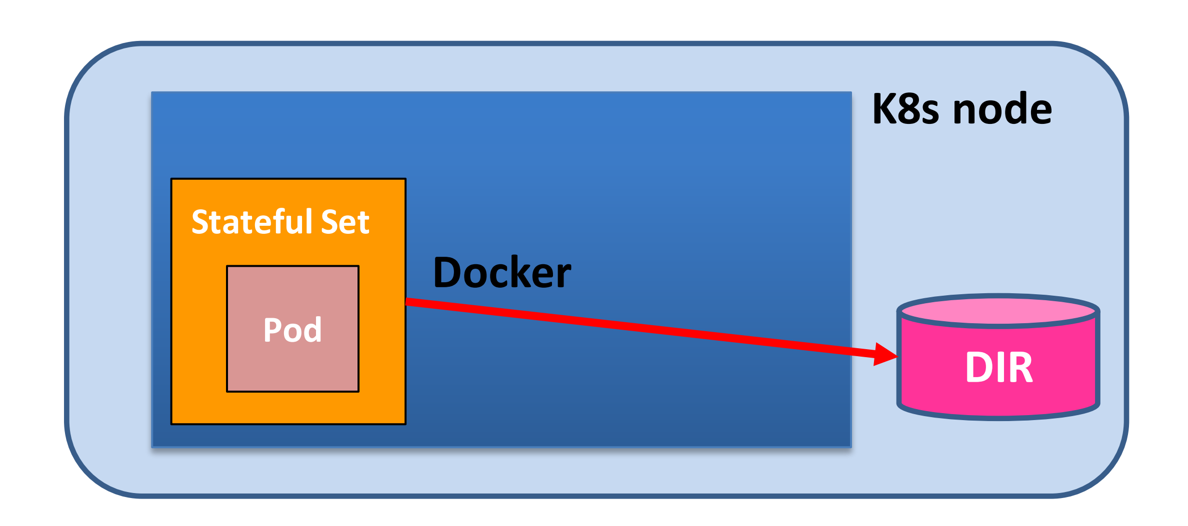


Persistent Storage – local storage

hostPath – /local/disk/path

+ performance

- complexity





Читаем документацию

for single node testing only; WILL NOT WORK in a multi-node cluster;



Persistent Storage – local storage

K8s node

Stateful Set

Pod

Docker

Node 1

DIR

with
data
files



Что происходит при смене Node, на которой исполняется Pod



Persistent Storage – local storage

K8s node

Stateful Set

Pod

Docker

DIR

NO DATA FILES

Node 2



Читаем документацию

WILL NOT WORK in a multi-node cluster;



Что будем делать?



Affinity & anti-affinity

Разложить Pods по нужным Nodes

Example from operator



```
 - name: clickhouse-per-host-on-servers-with-ssd
       spec:
         affinity:
           # Specify Pod affinity to nodes with specified properties (hosttype=ch-ssd)
           nodeAffinity:
             requiredDuringSchedulingIgnoredDuringExecution:
               nodeSelectorTerms:
                 - matchExpressions:
                     - key: “hosttype"
                       operator: In
                       values:
                         - “ch-ssd"
           # Specify Pod anti-affinity to Pods with the same label "/app" on the same
"hostname"
           podAntiAffinity:
             requiredDuringSchedulingIgnoredDuringExecution:
               - labelSelector:
                   matchExpressions:
                     - key: "clickhouse.altinity.com/app"
                       operator: In
                       values:
                         - "chop"
                 topologyKey: "kubernetes.io/hostname"
```

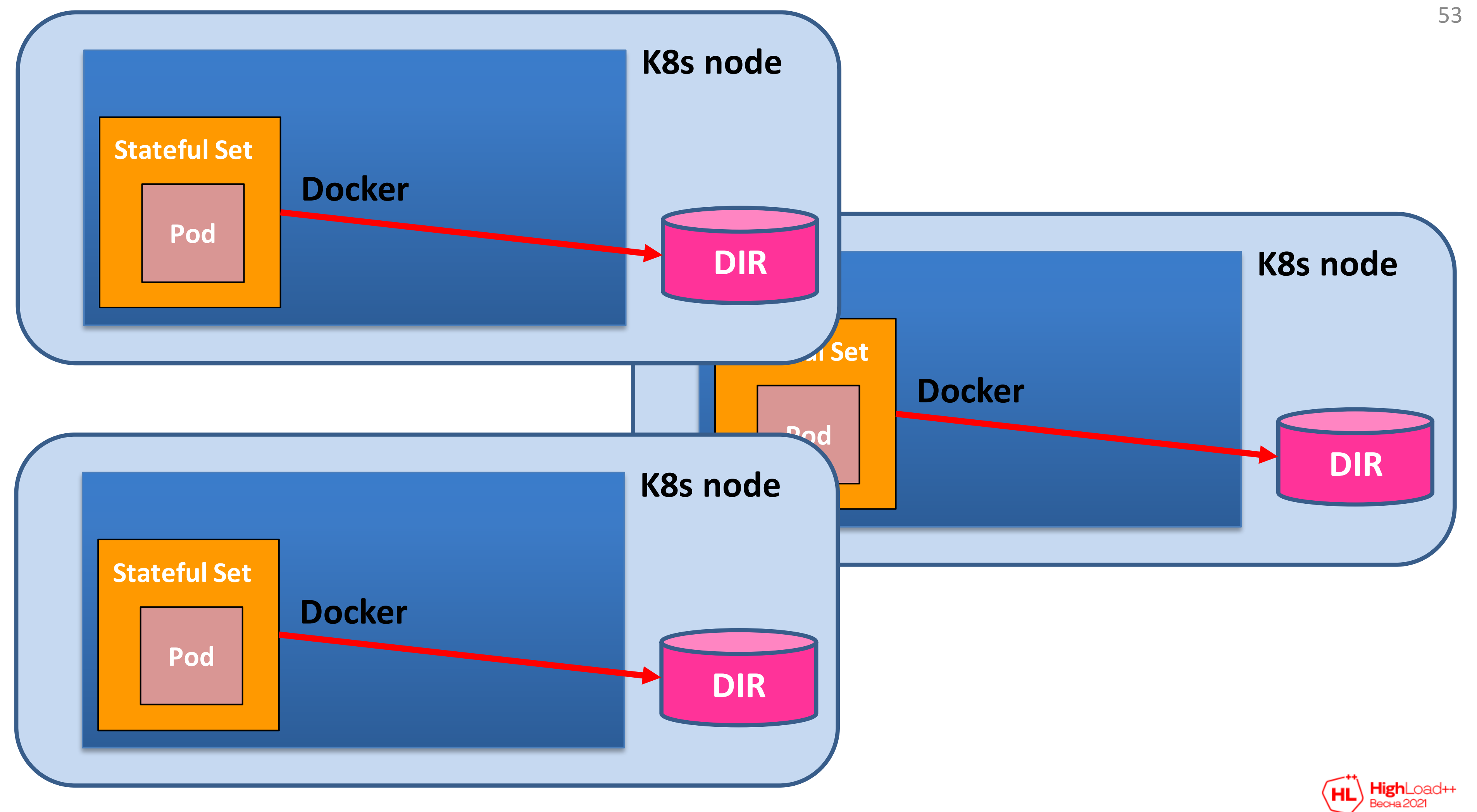


```
- name: clickhouse-per-host-on-servers-with-ssd
       spec:
         affinity:
           # Specify Pod affinity to nodes with specified properties (hosttype=ch-ssd)
           nodeAffinity:
             requiredDuringSchedulingIgnoredDuringExecution:
               nodeSelectorTerms:
                 - matchExpressions:
                     - key: “hosttype"
                       operator: In
                       values:
                         - “ch-ssd"
           # Specify Pod anti-affinity to Pods with the same label "/app" on the same
"hostname"
           podAntiAffinity:
             requiredDuringSchedulingIgnoredDuringExecution:
               - labelSelector:
                   matchExpressions:
                     - key: "clickhouse.altinity.com/app"
                       operator: In
                       values:
                         - "chop"
                 topologyKey: "kubernetes.io/hostname"
```



```
 - name: clickhouse-per-host-on-servers-with-ssd
        spec:
          affinity:
            # Specify Pod affinity to nodes with specified properties (hosttype=ch-ssd)
            nodeAffinity:
              requiredDuringSchedulingIgnoredDuringExecution:
                nodeSelectorTerms:
                  - matchExpressions:
                      - key: "hosttype"
                        operator: In
                        values:
                          - "ch-ssd"
            # Specify Pod anti-affinity to Pods with the same label "/app" on the same
"hostname"
            podAntiAffinity:
              requiredDuringSchedulingIgnoredDuringExecution:
                - labelSelector:
                    matchExpressions:
                      - key: "clickhouse.altinity.com/app"
                        operator: In
                        values:
                          - "chop"
                  topologyKey: "kubernetes.io/hostname"
```




K8s node
Stateful Set
Pod
Docker
DIR
K8s node
Set
Docker
DIR
K8s node
Stateful Set
Pod
Docker
DIR
HL
HighLoad++
Весна 2021

Указать локальную директорию



```
- name: clickhouse-per-host-on-servers-with-ssd
        spec:
            volumes:
          # Specify volume as path on local filesystem as a directory which will be created, if need be
          - name: local-path
            hostPath:
              path: /mnt/podvolume
              type: DirectoryOrCreate
        containers:
          - name: clickhouse-pod
            image: yandex/clickhouse-server:20.7
            volumeMounts:
              # Specify reference to volume on local filesystem
              - name: local-path
                mountPath: /var/lib/clickhouse
```

Восстановление данных:

Возможно в полном объёме

Они и не терялись



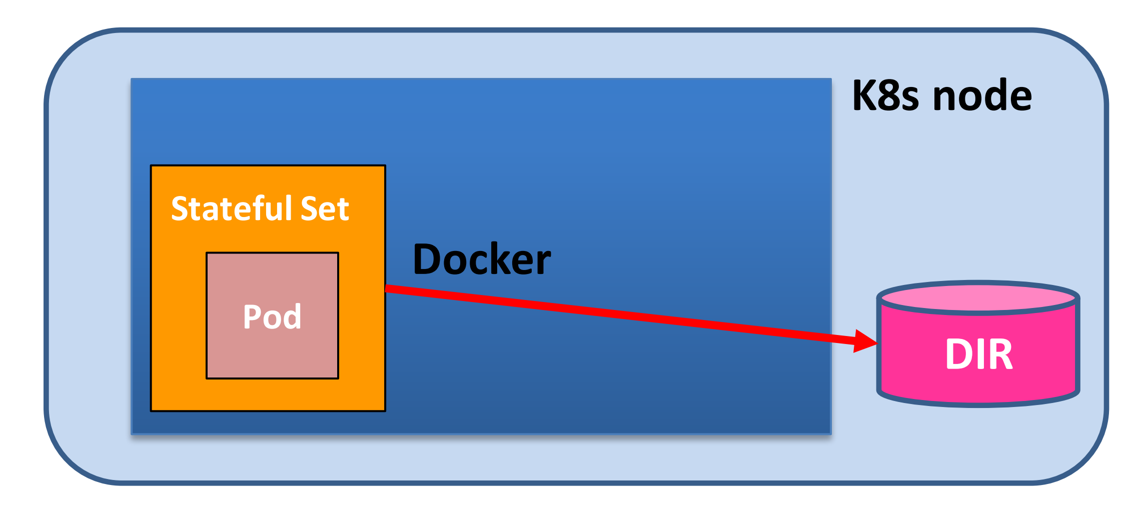
Развитие — Local Volume



Persistent Storage — local storage

Local - without manually scheduling Pods to nodes, as the system is aware of the volume's node constraints

- provisioning





Как так получается, что system is aware of the volume's node constraints?



А вот так

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: local-pv
spec:
local:
    path: /mnt/disks/ssd1
  nodeAffinity:
    required:
      nodeSelectorTerms:
      - matchExpressions:
        - key: kubernetes.io/hostname
          operator: In
          values:
          - node1
```



Читаем документацию

Local volumes can only be used as a statically created PersistentVolume. Dynamic provisioning is not supported.



Что будем делать?



Нужен Provisioner!



Часть 2. Automation



Provisioner

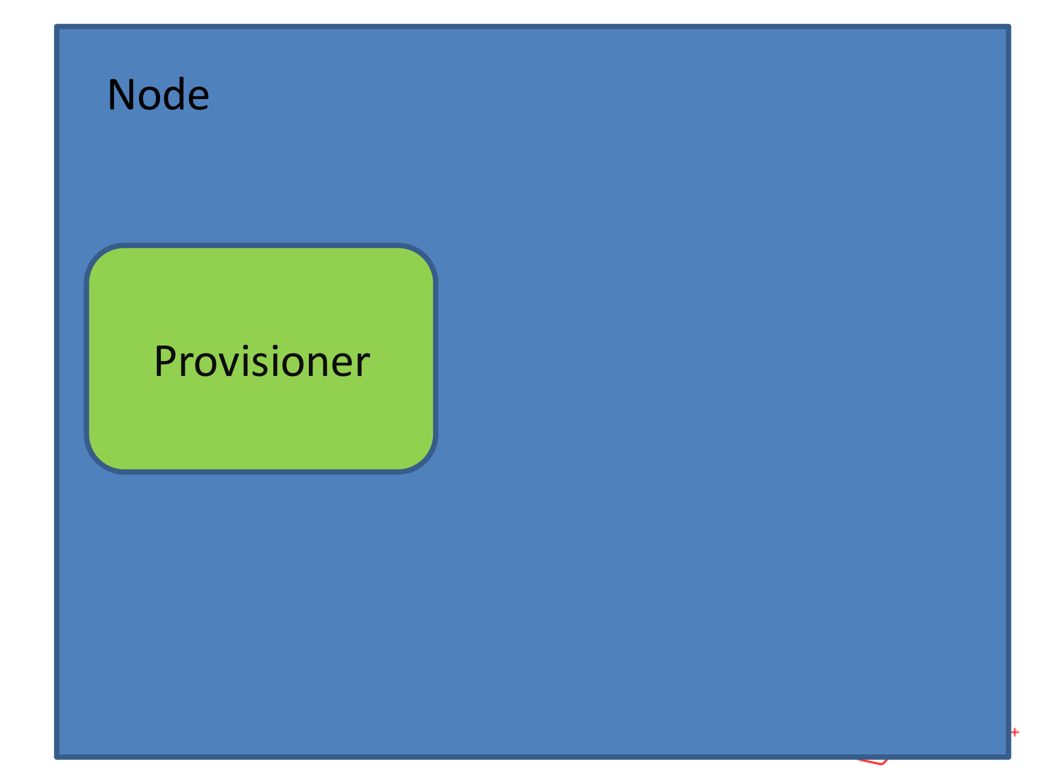
Independent program that follow a specification defined by Kubernetes. Authors of external provisioners have full discretion over where their code lives, how the provisioner is shipped, how it needs to be run



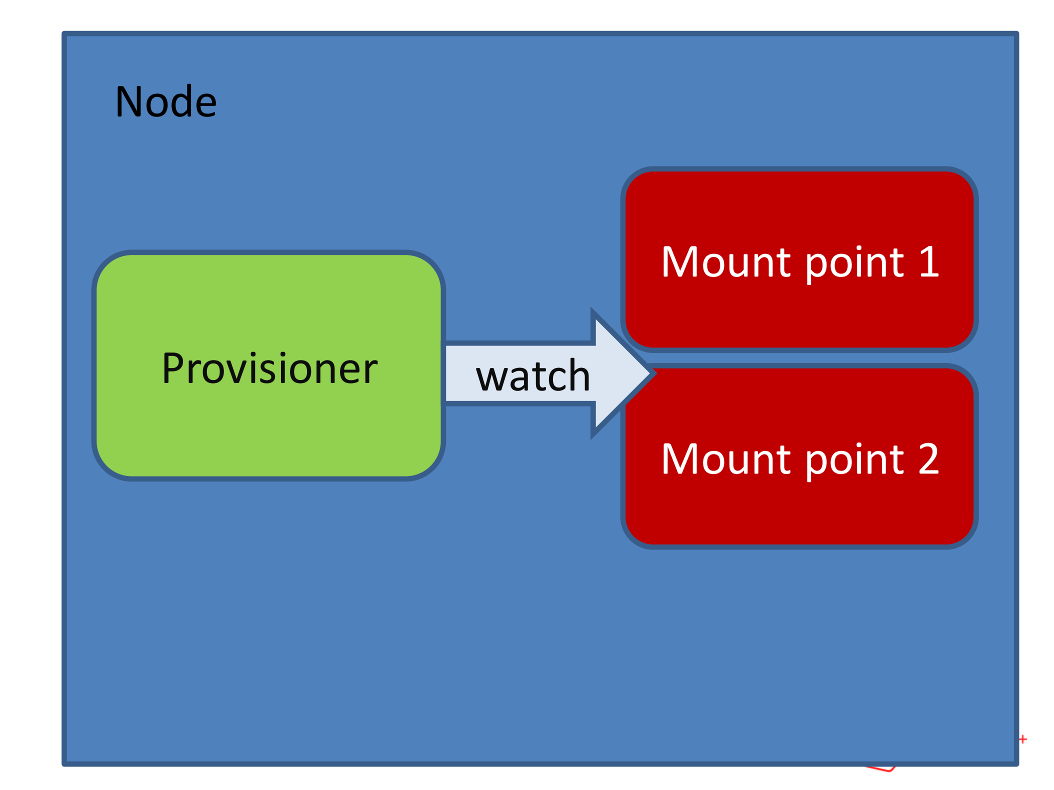
Static provisioning
Dynamic provisioning



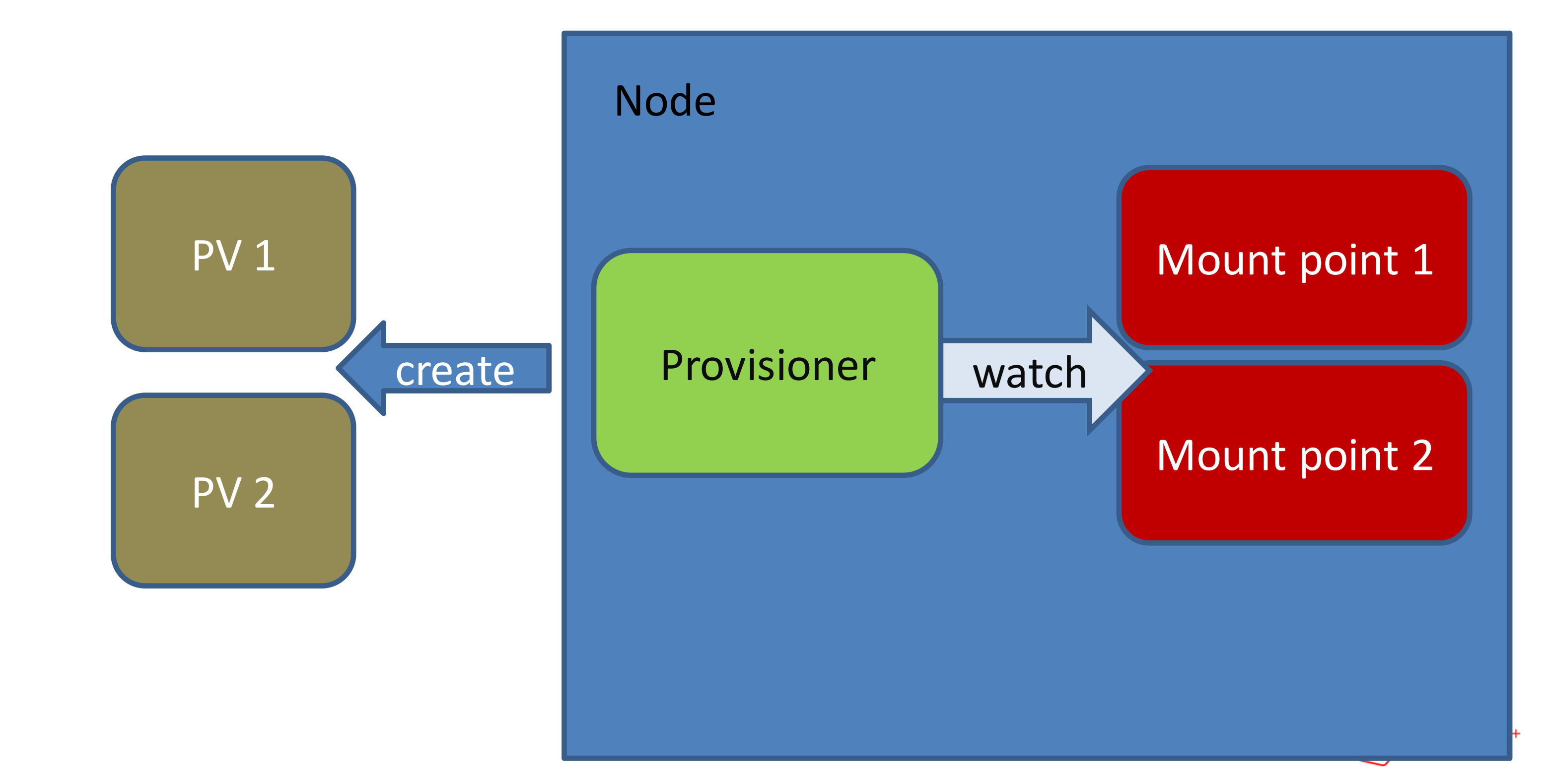
Автоматизация Static Provisioning



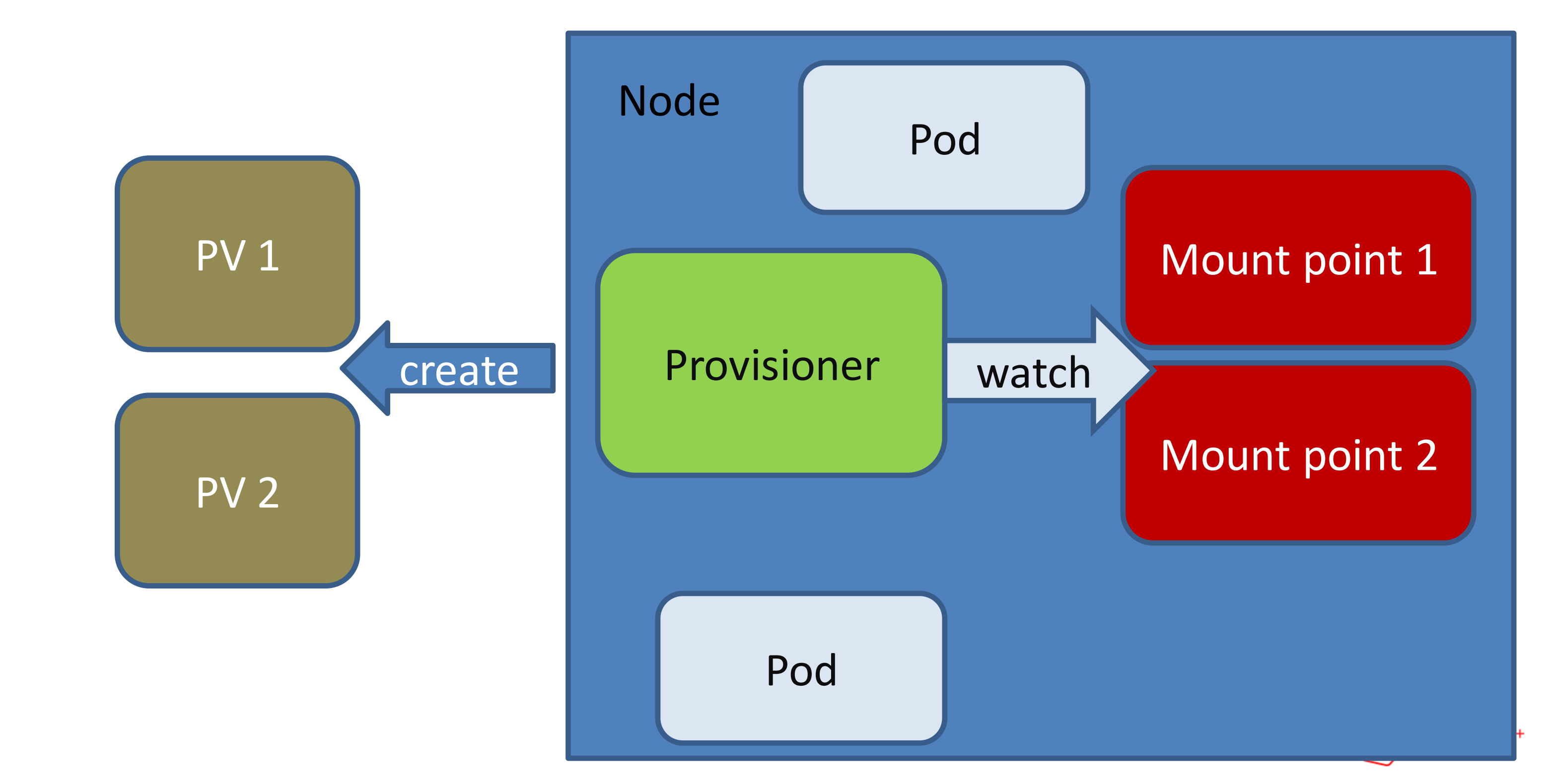
Автоматизация Static Provisioning



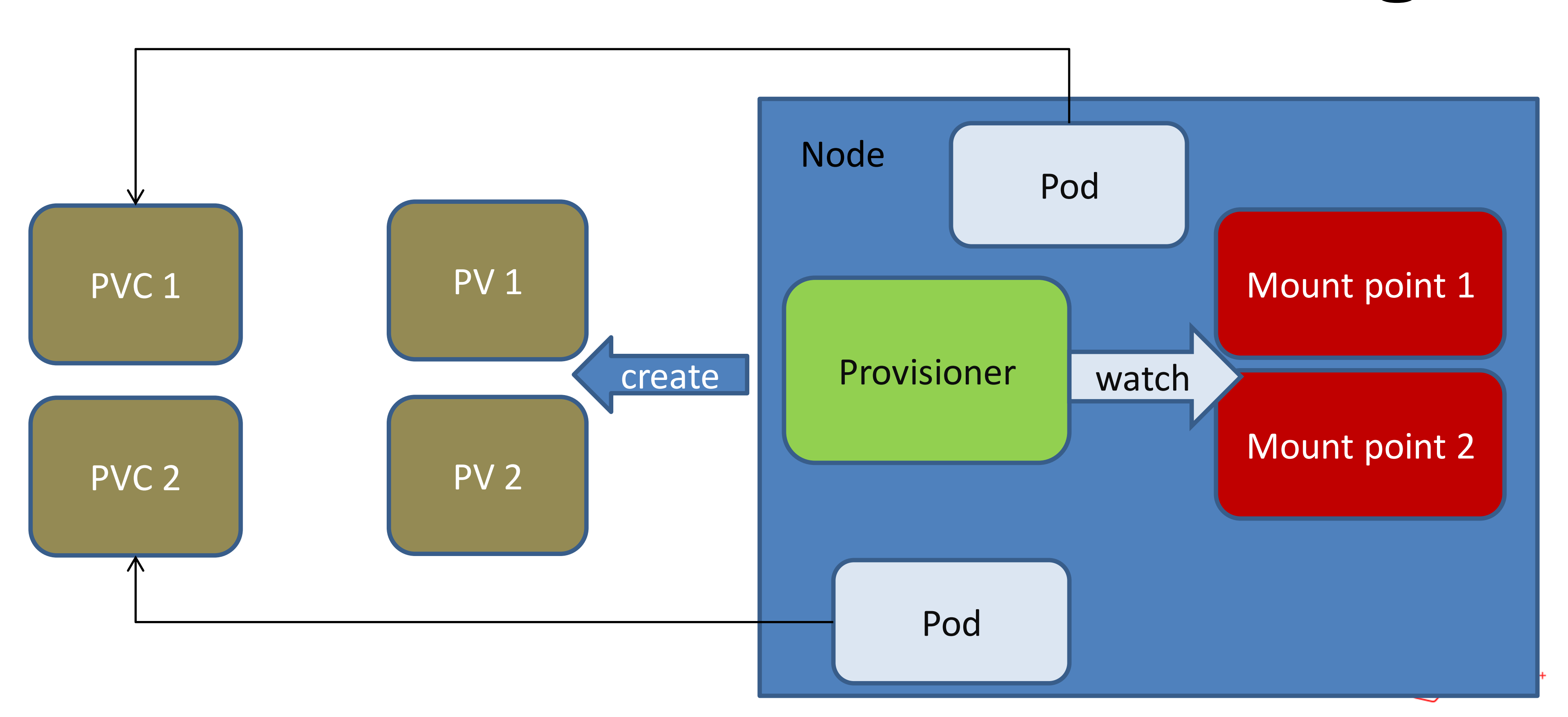
Автоматизация Static Provisioning



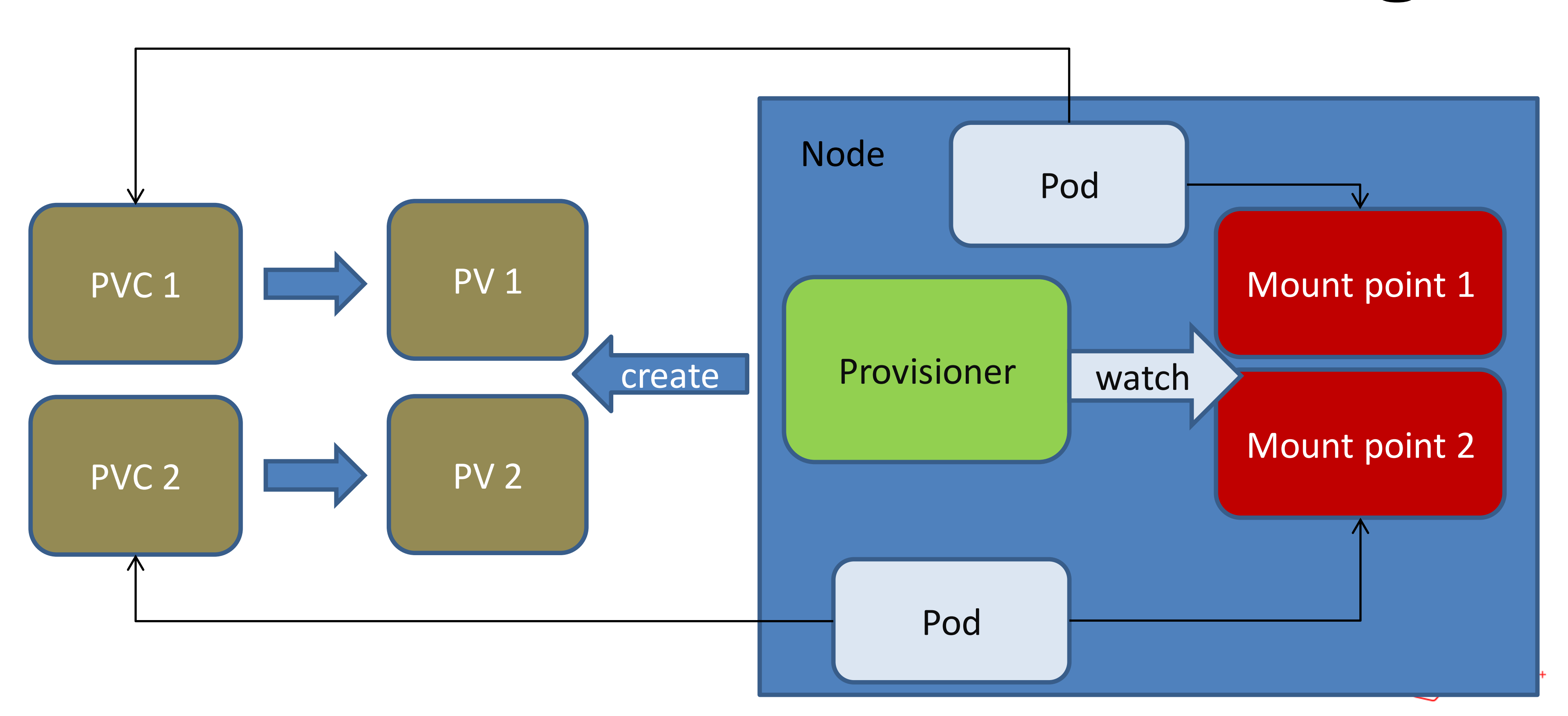
Автоматизация Static Provisioning



Автоматизация Static Provisioning



Автоматизация Static Provisioning



Автоматизация Static Provisioning

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: fast-disks
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: WaitForFirstConsumer
reclaimPolicy: Delete
```

https://github.com/kubernetes-sigs/sig-storage-local-static-provisioner



Читаем документацию

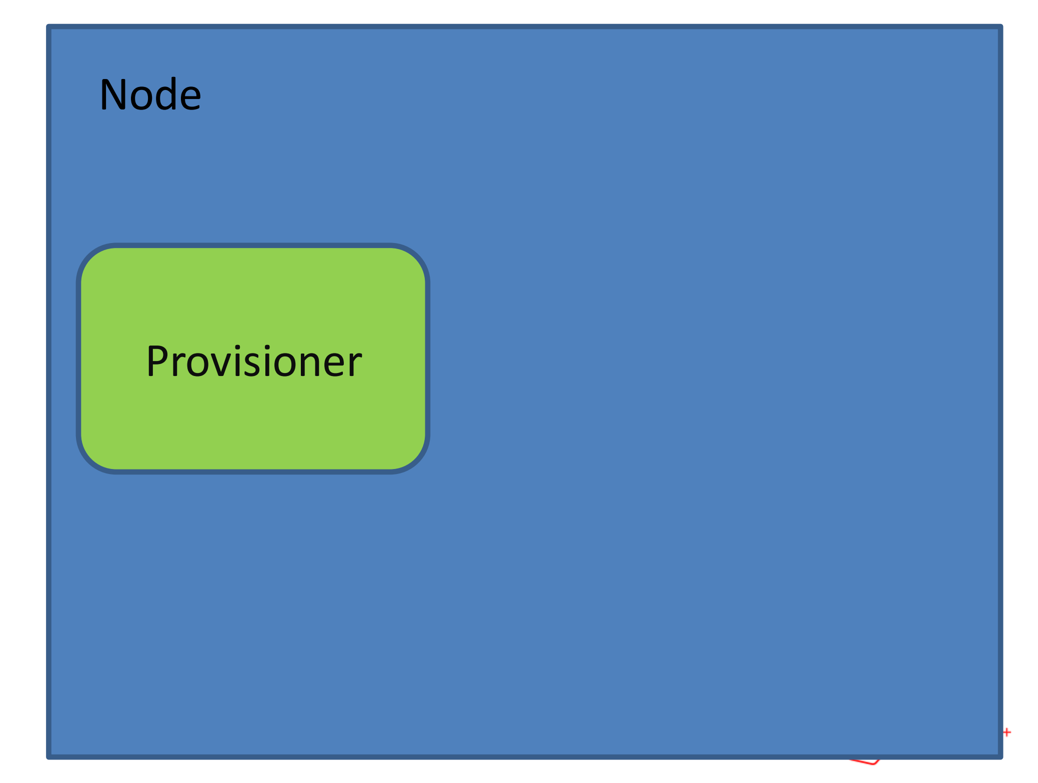
Local volumes can only be used as a statically created PersistentVolume. Dynamic provisioning is not supported.



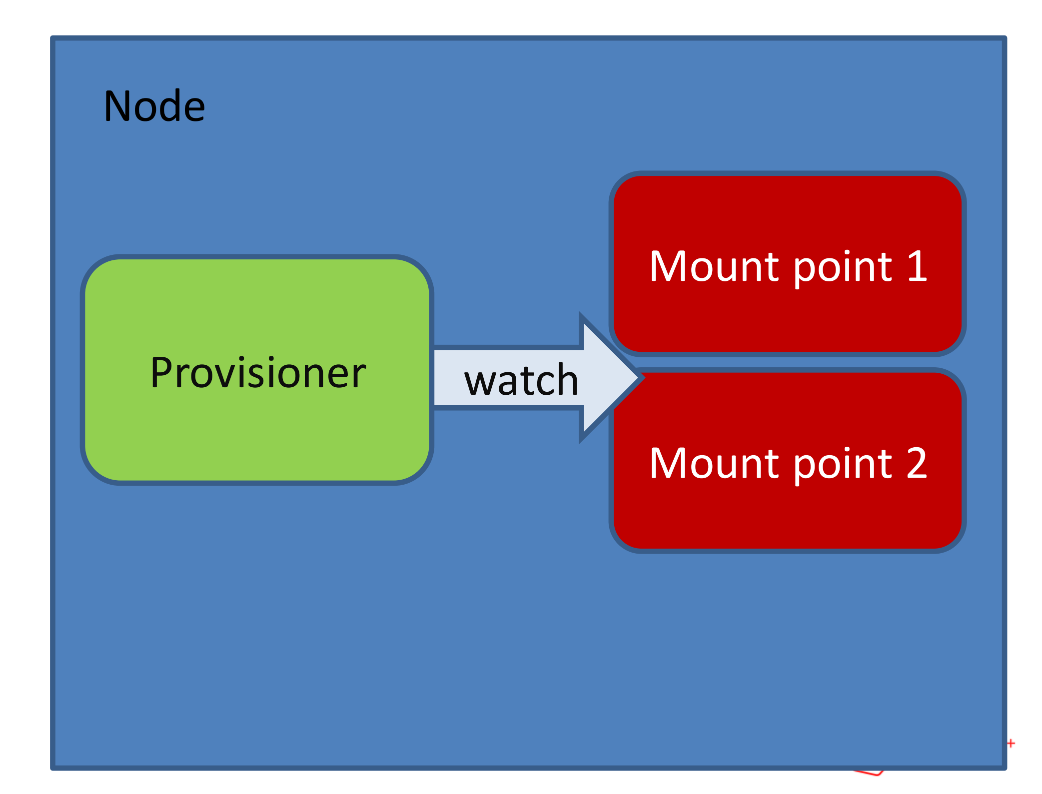
Вроде бы нельзя. Но очень хочется. Надо пробовать.



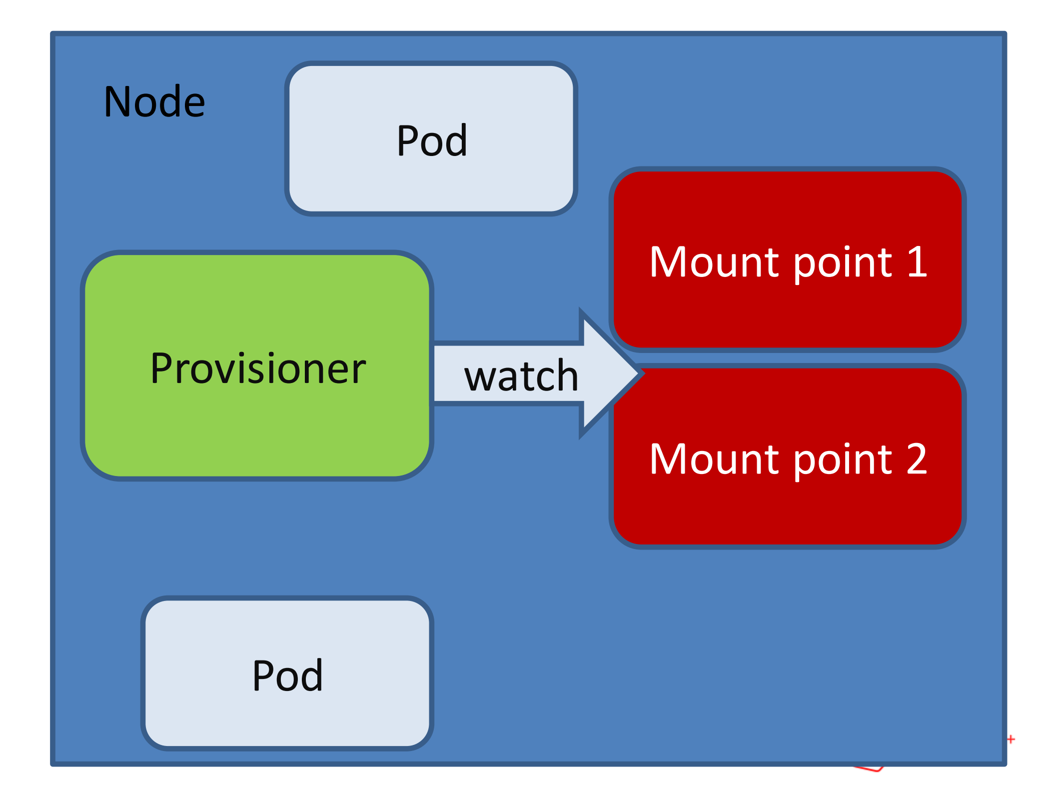
Dynamic Provisioning



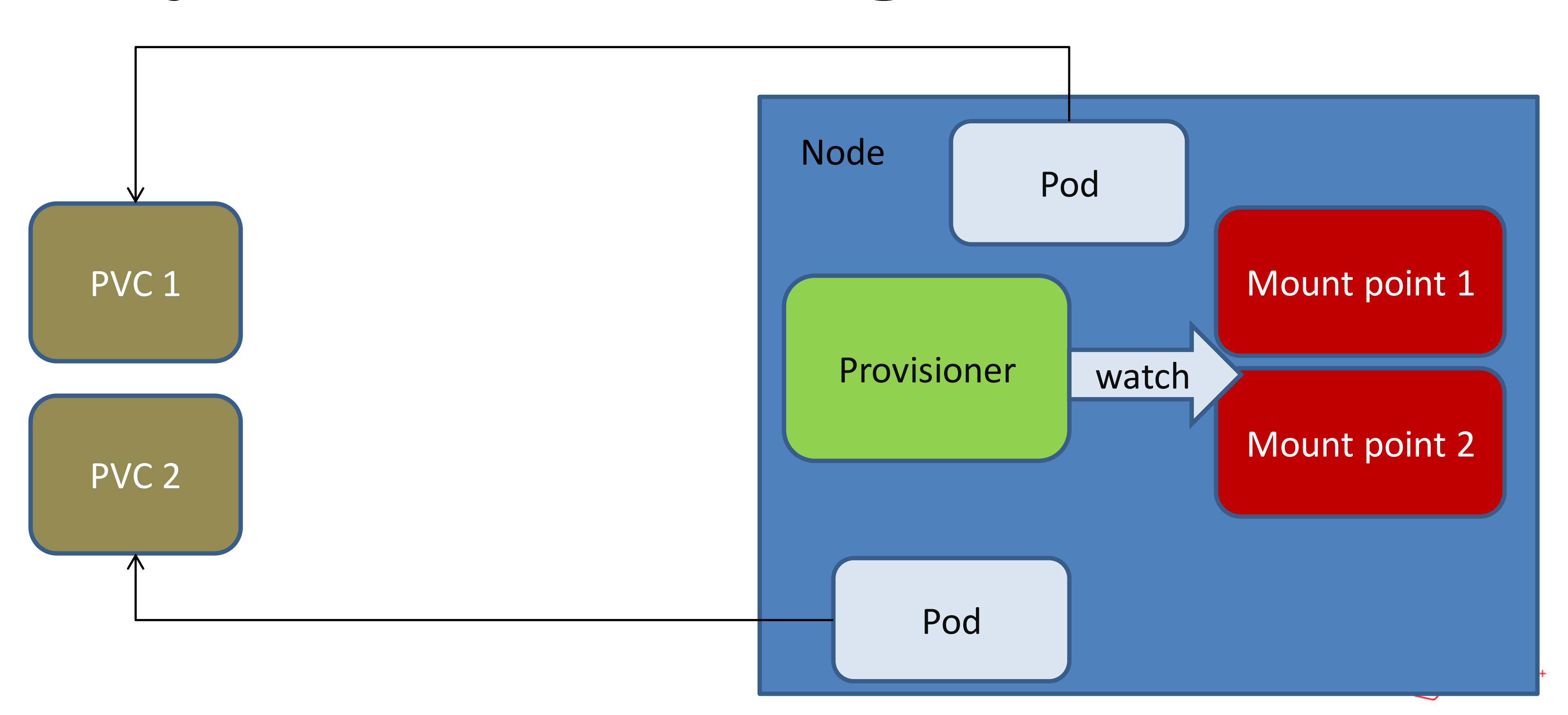
Dynamic Provisioning



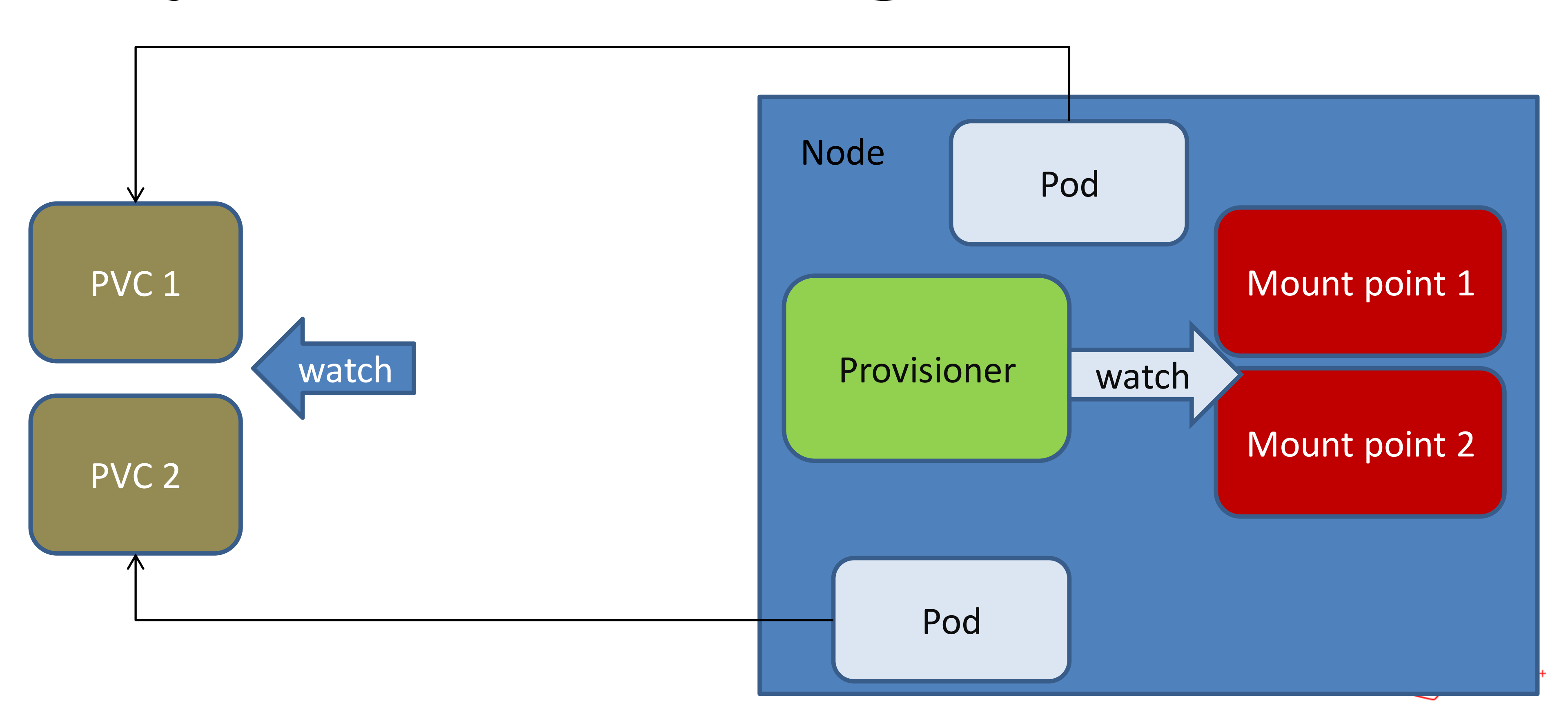
Dynamic Provisioning



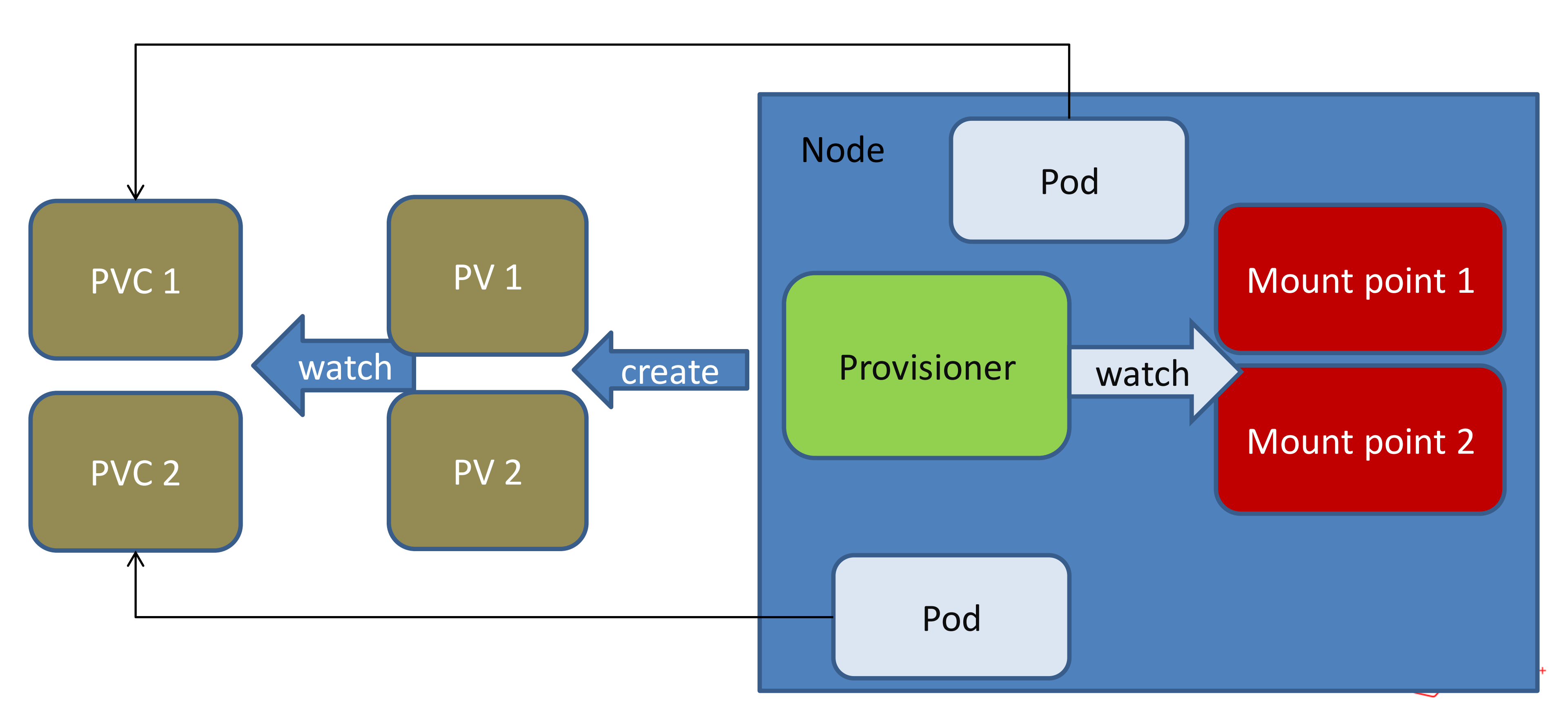
Dynamic Provisioning



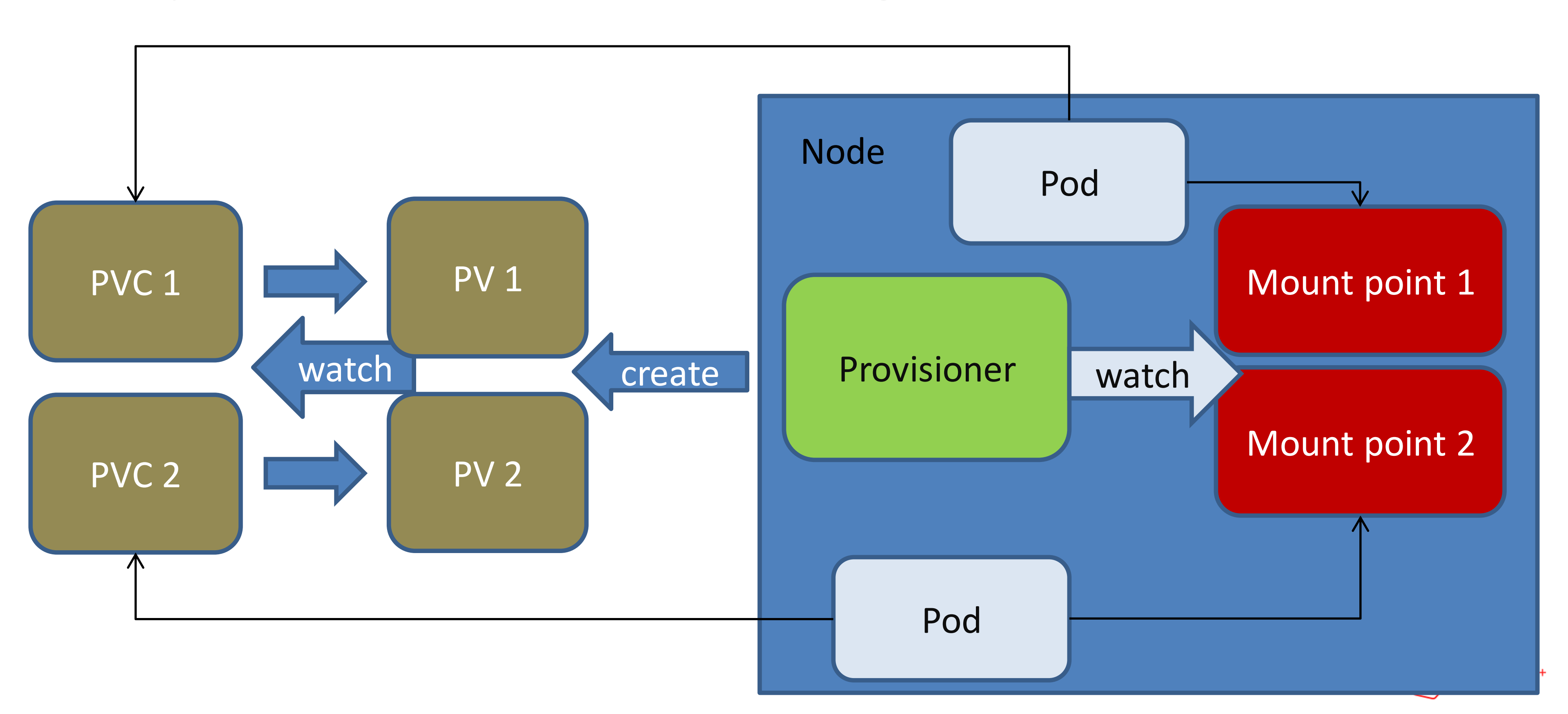
Dynamic Provisioning



Dynamic Provisioning



Dynamic Provisioning



Dynamic Provisioning

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: dynamic
provisioner: dynamic-local-provisioner
volumeBindingMode: WaitForFirstConsumer
reclaimPolicy: Delete
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: dynamic-local-pvc
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: dynamic
  resources:
    requests:
      storage: 128Mi
```

https://github.com/kubernetes-sigs/nfs-subdir-external-provisioner
https://github.com/rancher/local-path-provisioner/



lib

A library for writing external provisioners

https://github.com/kubernetes-sigs/sig-storage-lib-external-provisioner



Persistent Data в Kubernetes может быть даже с локальными дисками



ClickHouse Operator Github Project:

https://github.com/Altinity/clickhouse-operator

Issues & Pull Requests on Github



